ARMY RESEARCH LABORATORY

Smart Weapons Encounter Model

Richard J. Pearson
Kenrick K. Chien

ARL-TR-2178 APRIL 2000

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5066

ARL-TR-2178 April 2000

Smart Weapons Encounter Model

Richard J. Pearson
Kenrick K. Chien
Weapons & Materials Research Directorate

Abstract

This report covers the Smart Weapon Encounter Model (SWEM) developed to support the Tank Extended Range Munition (TERM) science and technology objective (STO) III G.3. The report decribes the model's algorithm, input, and output. SWEM uses solid geometry and statistical methods to calculate the probability that a target will fall within the field of view (FOV) of a sensor mounted on a smart weapon. When the target passes into the FOV of the sensor, an encounter is said to have occurred. SWEM calculates the probability of such encounters for a given sensor on a weapon traveling along a specific trajectory. The probability is determined by repeating a series of calculations in which the target's location or movements are randomly varied from calculation to calculation. Only the probability of encounter is determined. SWEM does not calculate the probability that the sensor system will be able to detect an encountered target against its background environment. Neither does it determine the probability that the weapon will be able to maneuver to a target once it is detected. However, SWEM does calculate the first necessary step in a series of steps leading to target interception.

ACKNOWLEDGMENTS

The authors would like to acknowledge the contribution of Mr. Patrick Hill who, while an employee of the U.S. Army Research Laboratory (ARL), conceived the encounter model as a tool for evaluating design concepts for the Tank Extended Range Munition. Mr. Hill delineated the basic requirements for the encounter model and helped guide its development.

The authors would also like to acknowledge the contribution of Dr. Joseph Wald, also of ARL. Software developed previously by Dr. Wald was reused and forms an important part of the Smart Weapon Encounter Model.

INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

INTENTIONALLY LEFT BLANK

LIST OF FIGURES

INTENTIONALLY LEFT BLANK

SMART WEAPONS ENCOUNTER MODEL

1. INTRODUCTION

The Smart Weapons Encounter Model (SWEM) calculates the probability that a sensor mounted on a projectile will encounter a target. The encounter occurs when the field of view (FOV) of the sensor moves over a target. In SWEM, a sensor's FOV is represented by a cone traveling in three-dimensional (3-D) space. SWEM calculates the probability of such an encounter occurring for a given set of sensors on a weapon traveling along a specific trajectory. The probability is determined by repeating a series of calculations in which the target's location or movements are randomly varied from calculation to calculation. The range of the random variation is determined by the performance characteristics of the projectiles, the sensors, and the target vehicle.

SWEM calculates only the probability of encounter. An encounter must occur before a target can be detected or the weapon can maneuver to the target. SWEM does not calculate the probability that the sensor system will able to detect a target against the background environment. Neither will SWEM determine if a weapon will be able to maneuver to a target that has been detected. However, SWEM does calculate the first necessary step in the process of intercepting a target.

SWEM was developed to support the Tank Extended Range Munition (TERM) science and technology objective (STO) III G.3. In the TERM concept, a smart projectile is fired from the main armament of existing tanks. After launch, the TERM projectile follows a ballistic trajectory or maneuvers along a predetermined path while its sensors look for a target. If a target is encountered and detected, the projectile autonomously maneuvers to intercept it.

TERM can operate beyond line of sight of the firing tank. Information from land-based scouts, helicopters, or unmanned aerial vehicles (UAVs) is used to locate targets. The information is used to calculate an aim point for the TERM projectile. The TERM projectile flies to the aim point and attempts to detect a target along the way. If a target is detected, the TERM projectile attempts to maneuver to intercept it.

The accuracy of the scout's target observations, the time required to relay information to the firing tank, and the delivery accuracy of the TERM round determine how close the round will pass to the target. How close the projectile comes to the target during the search phase is an important factor in determining the probability of encounter. The maximum range of the sensors and the size of their FOVs are also important factors in determining the probability of encounter. In addition, the path flown by the projectile while it tries to detect the target will influence the probability of encounter. SWEM considers all these factors in its calculations.

2. THE GEOMETRY OF A SENSOR ENCOUNTER

As stated in Section 1, the view of a sensor mounted on a smart weapon can be modeled as a cone moving in 3-D space. In SWEM, the FOV cross section is assumed to be elliptical and is defined by two viewing angles and the distance from the sensor. The length of the FOV is determined by the maximum range of the sensor. Figure 1 shows the FOV of a sensor in *projectile coordinates* (XP, YP, ZP).

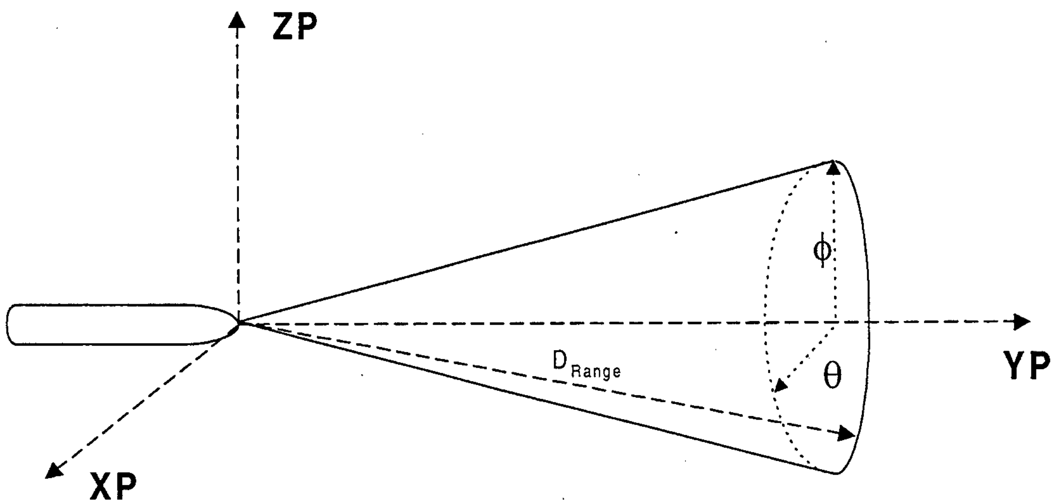

Figure 1. Sensor FOV.

In Figure 1, the sensor is mounted in the nose of the projectile and is aligned with the long axis. The projectile is flying horizontally to the right. The Y-axis is aligned with the center of the sensor's FOV. The positive Z direction is defined as "up." The X-axis is normal to the Y-Z plain. Theta (θ) is the sensor viewing half angle in azimuth. Phi (ϕ) is the sensor viewing half angle in elevation. The equation for the surface of the FOV is as follows:

$$\frac{XP_{target}^{\;2}}{\left(YP_{target}^{\;2}\cdot\tan(\theta)^2\right)}+\frac{ZP_{target}^{\;2}}{\left(YP_{target}^{\;2}\cdot\tan(\phi)^2\right)}=V_{Ratio} \qquad \text{Eq. 1}$$

In Equation 1, XP_{target}, YP_{target}, and ZP_{target} define the location of a target. For targets inside the FOV, V_{Ratio} is less than 1. For target points outside the FOV, V_{Ratio} is greater than 1. If V_{Ratio} is equal to 1, the target is exactly on the surface that defines the boundary of the FOV.

Target points behind the projectile, as well as those in front, will produce V_{Ratio}s less than 1. Only forward-looking sensors are considered in the SWEM code, however. Because only forward-looking sensors are investigated, the target points that have a V_{Ratio} less than 1 and are behind the projectile are ignored.

The other factor that must be considered in determining which points are within the FOV of the sensor is the sensor's maximum range. The distance from the sensor to any target point around it is calculated as follows:

$$D_{Range} = \left[\left(XP_{target}\right)^2 + \left(YP_{target}\right)^2 + \left(ZP_{target}\right)^2 \right]^{\frac{1}{2}} \qquad \text{Eq. 2}$$

If the distance to a target D_{Range} is less than the maximum range of a sensor and the V_{Ratio} of the target point is ≤ 1, then the target is encountered by the sensor. For the coordinate system aligned and moving with the projectile, Equations 1 and 2 define an encounter with a target.

It is possible to define the movement of the targets in terms of the coordinate system shown in Figure 1 where the projectile is stationary. It is easier, however, to transform Equations 1 and 2 to a coordinate system where the projectile moves relative to the earth. Figure 2 shows the projectile in such a coordinate system.

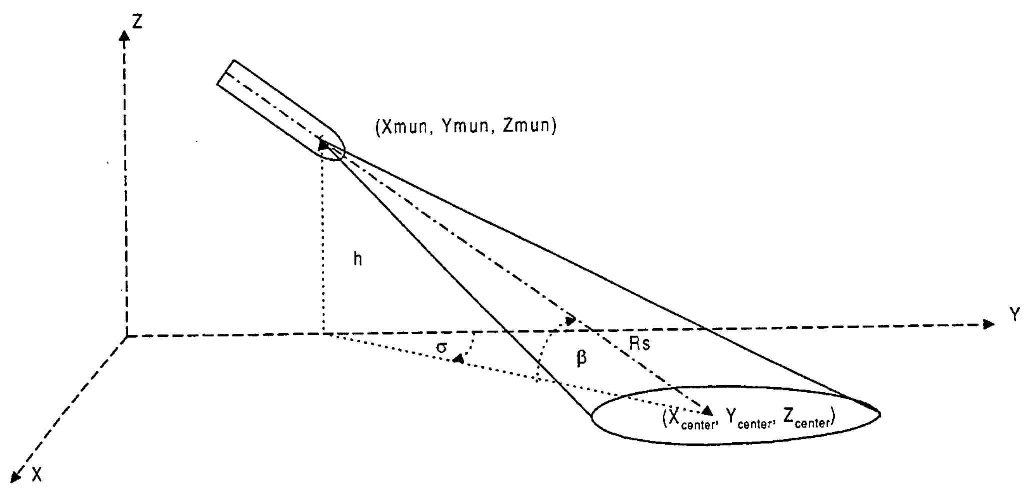

Figure 2. TERM Projectile in the Trajectory Coordinate System.

The coordinate system shown in Figure 2 is called the "*trajectory coordinate system*" and has its origin at the position of the firing tank. In the *trajectory coordinate system*, the Z-axis is aligned with local vertical. Positive Y is in the down-range direction. The Y-Z plane contains the velocity vector of the projectile at launch. Positive X is in the cross-range direction and to the right. The X-axis is normal to the Y-Z plane.

The position of the projectile in this *trajectory coordinate system* is given as X_{mun}, Y_{mun}, and Z_{mun}. Euler angles are used to specify the orientation of the projectile in the *trajectory coordinate system*. The yaw angle sigma (σ) is measured from the Y-Z plane to the long axis of the projectile. The pitch angle beta (β) is measured from the X-Y plane to the long axis of the projectile. The roll angle is not used because the projectile is not assumed to roll in flight.

Transforming from the *projectile coordinate system* to the *trajectory coordinate system* involves a translation and two rotations. The Euler angle convention that is used is defined in *Methods of Analytical Dynamics* (Meirovitch 1970). The transformation using the Euler angles is defined as three successive rotations. The first rotation is in yaw about the Z-axis. The second rotation is in pitch about the X-axis. The third rotation, not used in this code, is in roll about the Y-axis.

The three Euler angles are used to define a matrix, which transforms the X, Y, and Z in the *trajectory coordinate system* into XP, YP, and ZP in the *projectile coordinate system*. The nine elements of the matrix are as follow:

$$B_{11}=\cos(\Theta_1)\cdot\cos(\Theta_3)-\sin(\Theta_1)\cdot\cos(\Theta_2)\cdot\sin(\Theta_3)$$

$$B_{12}=\sin(\Theta_1)\cdot\cos(\Theta_3)+\cos(\Theta_1)\cdot\cos(\Theta_2)\cdot\sin(\Theta_3)$$

$$B_{13}=\sin(\Theta_1)\cdot\sin(\Theta_3)$$

$$B_{21}=-\cos(\Theta_1)\cdot\sin(\Theta_3)-\sin(\Theta_1)\cdot\cos(\Theta_2)\cdot\cos(\Theta_3)$$

$$B_{22}=-\sin(\Theta_1)\cdot\sin(\Theta_3)+\cos(\Theta_1)\cdot\cos(\Theta_2)\cdot\cos(\Theta_3)$$

$$B_{23}=\sin(\Theta_1)\cdot\cos(\Theta_3)$$

$$B_{31}=\sin(\Theta_1)\cdot\sin(\Theta_2)$$

$$B_{32}=-\cos(\Theta_1)\cdot\sin(\Theta_2)$$

$$B_{33}=\cos(\Theta_2) \qquad \text{Eq. 3}$$

In Equation 3, Θ_1, Θ_2, and Θ_3 are the yaw, pitch, and roll angles, respectively. In SWEM, $\Theta_1 = -\sigma$, $\Theta_2 = \beta$, and $\Theta_3 = 0$. Substituting for yaw, pitch, and roll angles and applying the matrix, the XP_{target}, YP_{target}, and ZP_{target} can be written in terms of the projectile position (X_{mun}, Y_{mun}, and Z_{mun}) and the target location (X_{target}, Y_{target}, Z_{target}) in *trajectory coordinates* as follows:

$$XP_{target}=\cos(-\sigma)\cdot X_{target}+\sin(-\sigma)\cdot(Y_{target}-Y_{mun})$$

$$YP_{target}=-\sin(-\sigma)\cdot\cos(\beta)\cdot(X_{target}-X_{mun})+\cos(-\sigma)\cdot\cos(\beta)\cdot(Y_{target}-Y_{mun})+\sin(-\sigma)\cdot\cos(\beta)\cdot(Z_{target}-Z_{mun})$$

$$ZP_{target}=\sin(-\sigma)\cdot\sin(\beta)\cdot(X_{target}-X_{mun})-\cos(-\sigma)\cdot\sin(\beta)\cdot(Y_{target}-Y_{mun})+\cos(\beta)\cdot(Z_{target}-Z_{mun}) \qquad \text{Eq. 4}$$

Substituting for XP_{target}, YP_{target}, and ZP_{target} in Equation 1, the V_{ratio} can be written in terms X_{target}, Y_{target}, Z_{target}, X_{mun}, Y_{mun}, Z_{mun}, β and σ. The results of the substitution are the following:

$$R_{\theta} + R_{\phi} = V_{Ratio} \qquad \text{Eq. 5}$$

in which

$$R_{\theta} = \frac{\left[\cos(-\sigma)\cdot\left(X_{target} - X_{mun}\right) + \sin(-\sigma)\cdot\left(Y_{target} - Y_{mun}\right)\right]^2}{\left[\left[-\sin(-\sigma)\cdot\cos(\beta)\cdot\left(X_{target} - X_{mun}\right) + \cos(-\sigma)\cdot\cos(\beta)\cdot\left(Y_{target} - Y_{mun}\right) + \sin(-\sigma)\cdot\cos(\beta)\cdot\left(Z_{target} - Z_{mun}\right)\right]^2\cdot\tan(\theta)^2\right]}$$

$$R_{\phi} = \frac{\left[\sin(-\sigma)\cdot\sin(\beta)\cdot\left(X_{target} - X_{mun}\right) - \cos(-\sigma)\cdot\sin(\beta)\cdot\left(Y_{target} - Y_{mun}\right) + \cos(\beta)\cdot\left(Z_{target} - Z_{mun}\right)\right]^2}{\left[\left[-\sin(-\sigma)\cdot\cos(\beta)\cdot\left(X_{target} - X_{mun}\right) + \cos(-\sigma)\cdot\cos(\beta)\cdot\left(Y_{target} - Y_{mun}\right) + \sin(-\sigma)\cdot\cos(\beta)\cdot\left(Z_{target} - Z_{mun}\right)\right]^2\cdot\tan(\phi)^2\right]}$$

The distance between the center of the sensor and the center of the target is the parameter calculated by Equation 2. Because Equation 2 defines the length or magnitude of a vector, it is not affected by rotations in the transformation. Applying the translation to Equation 2 gives the following:

$$D_{Range} = \left[\left(X_{target} - X_{mun}\right)^2 + \left(Y_{target} - Y_{mun}\right)^2 + \left(Z_{target} - Z_{mun}\right)^2\right]^{\frac{1}{2}} \qquad \text{Eq. 6}$$

Equations 5 and 6 form the basic algorithm, which is used to determine if a target is encountered. The algorithm determines if a target located at (X_{target} Y_{target} Z_{target}) is within the FOV of the sensor. As input, the algorithm takes target location, projectile position (X_{mun}, Y_{mun}, Z_{mun}), the projectile orientation (β, σ), and the view angles of the sensor (θ, ϕ). The algorithm outputs a Boolean value (true or false) as to whether the target is within the FOV and therefore encountered.

In the SWEM code, the position and orientation of the projectile are read from a trajectory file. Four different trajectory file formats can be used by the SWEM code. An example of the standard trajectory format is shown in Appendix A. The code reads the files and interpolates between points to produce parameters at even time steps.

3. STATIONARY TARGETS

The SWEM code can handle either stationary targets or moving targets. To calculate the probability of encounter, SWEM uses a "Monte Carlo" method. The code loops through the same simulation a large number of times; each pass is termed a "replication." At the start of each replication, a set of random number draws is performed to obtain values for parameters governing the motion and/or location of the target. The projectile trajectory file and sensor parameters remain the same in every replication. The replications that result in a target encounter are summed at the end of the loop and are compared to the total number of replications to determine the probability of encounter.

For stationary targets, the probability of encounter depends on two types of errors: weapon dispersion error (WDE) and target location error (TLE). The WDE is a reflection of the weapon system's ability to hit the point at which it is aimed. The TLE reflects the ability

of the system to determine the location of the target in some coordinate system common to both the scout vehicle and firing tank.

Both WDEs and TLEs are decomposed into range and deflection errors. Range refers to an error in the length of the vector connecting the target to the scout or firing tank. The deflection errors are measured at right angles to the range vector. In the SWEM code, these errors are defined via a mean and standard deviation.

The TLE is composed of several separate errors in real systems. Self-location error refers to the ability of the scout vehicle and the firing tank to determine their own locations in space. Range error refers to the ability of the scout to determine the distance to the target. There are also errors in determining the heading of the scout and firing tank, called north-finding error. There are additional errors in finding the local direction of gravity. For the scout, there are additional errors in the measurement of the azimuth and elevation of the target relative to north and the local vertical. The self-location, range, north-finding, gravity, azimuth, and elevation errors are all combined into the target location range and deflection errors which are input to the SWEM code. Methods for combining the individual errors into TLEs required by SWEM are covered in the *Target Location Error Methodology for the Tank Extended Range Munition* (Wald 1997).

A FORTRAN (Formula Translator) subroutine "*rndm*" is employed to make random draws from a statistical distribution. The subroutine was adapted from an existing code and reused (Wald 1999). Four random draws are performed for the delivery range error, the delivery deflection error, the target location range error, and the target location deflection error.

The target begins at the aim point of the weapon in the *trajectory coordinate system*. Then, a randomly drawn delivery range error and a randomly drawn target location range error are added to the Y location of the target. The delivery deflection error and the target location deflection errors are added to the X location of the target. The resulting perturbed target position is used in the next simulation replication. The process is repeated at the start of each replication.

4. MOVING TARGETS

For moving targets, the target's path needs to be randomly varied between replications. The code used to calculate target paths was taken from Wald (1997). The FORTRAN code used is called "*makpth*". In *makpth*, the target path is composed of a number of "legs" in which the speed and direction of the target are constant. The code randomly varies the initial target location, the duration of each leg, the heading in each leg, the speed in each leg, and the average heading tendency for the whole path. The variation in the target path is determined by a set of 21 input parameters. The parameters and the input file that contains them are described in Section 7.

The *makpth* code calculates the position of the target at any time in its own *target path coordinates*. The origin of the *target path coordinate* system is located at or near the starting point of the target. The positive X direction is east and positive Y is north.

The problem of predicting the path of a moving target, based on observations of a remote scout, is shown in Figure 3 in which the scout takes two observations of the target tank. Linear prediction is used to estimate the future position of the target.

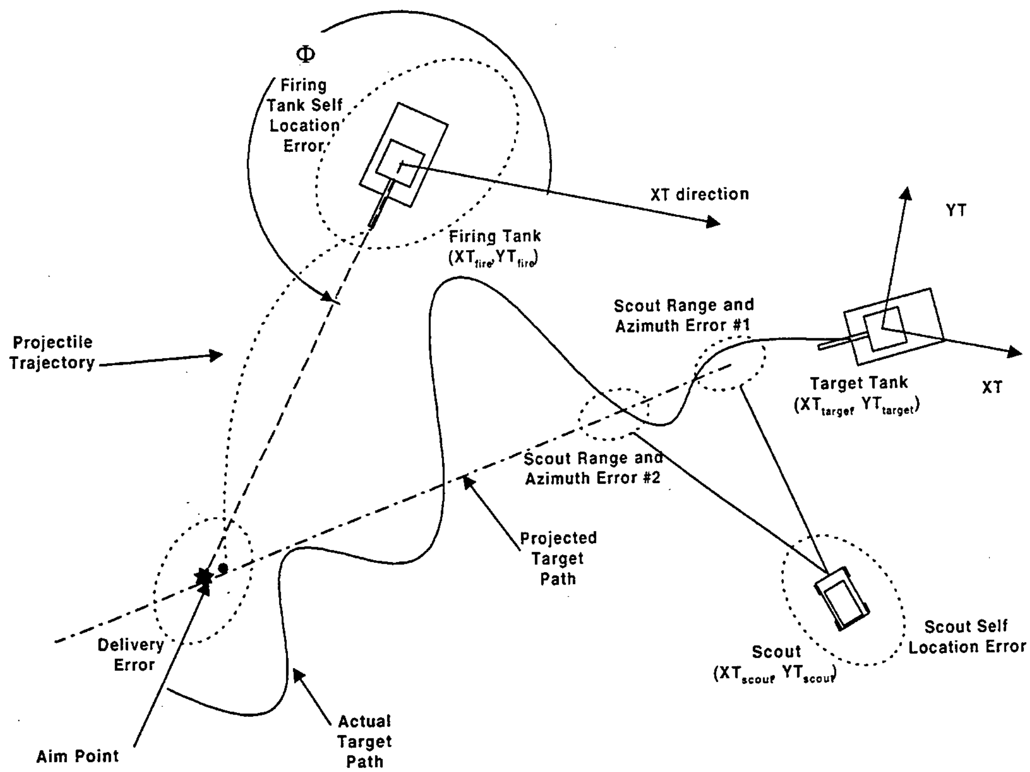

Figure 3. Predicting Target Path.

The types of errors in the two observations of the target made by the scout were discussed in Section 3. The scout observations have self-location range, north-finding, gravity, azimuth, and elevation errors. These errors all contribute to the error in estimating the velocity of the target. The velocity estimation errors and changes in target motion after the last scout observation add to the final TLE. The longer the period between the last observation and the arrival of the weapon at its aim point, the greater the error caused by the target motion. The errors associated with the firing tank are the same as in the stationary target case and again contribute to the total TLE. The dispersion error is unchanged by target motion. Wald (1997) describes in greater detail how TLE is calculated for moving targets.

The calculations of the total TLE for a moving target and the calculation of the target path share some parameters. The SWEM code does not check to see that these parameters

agree. Using the methods in Wald (1997), the user must ensure that the parameters are properly matched.

The time line for a smart weapon encounter with a moving target is governed by a number of factors. In SWEM, the time of the first scout observation after the target path begins is selected by a random draw calculated by the *rndm* subroutine. The time between the two observations is an input to SWEM. The time between the second observation and the launch of the projectile, called the "communication, decision, and implementation time," is also input. The time from launch to the arrival of the projectile at the aim point is determined from the trajectory file. Adding the time of the first observation, the time between observations, the "communication, decision, and implementation time," and the time of flight of the projectile gives the time along the target path at which the projectile arrives at its aim point in *target path* time.

Time zero in *trajectory time* occurs at projectile launch. The difference between *trajectory time* and *target path* time is the time of the first observation plus time between observations plus observation, communication, decision, and implementation time. To bring the *target path* time into alignment with *trajectory time*, the difference is subtracted from all times in the target path array. After the adjustment, the target starts moving at some negative time before projectile launch.

The transformation of the target path into the *trajectory coordinate system* is shown in Figure 4. The *target path* and *trajectory coordinate system* are related by the position of the firing tank and the position of the projectile's aim point. The firing tank's position (XT_{fire}, YT_{fire}) in the *target path* coordinate system is an input to SWEM. In the *trajectory coordinate system*, the firing tank is located at the origin (0,0). The position of the aim point is located at the end of the projectile's trajectory defined in *trajectory coordinates.*

The aim point is also located on the target path defined in *target path* coordinates. The aim point's position along the target path depends on the time of arrival of the projectile at the aim point in *target path* time. Using the position of the firing tank and the aim point in the two coordinate systems, one can define a translation and a rotation that transform the *target path* into the *trajectory coordinate system.*

Interpolation is used on the X versus time and the Y versus time arrays of the tank path to find the coordinates of the weapon's aim point (XT_{aim}, YT_{aim}). The angle between the X axis of the *target path* coordinate system and the vector connecting the aim point and firing tank is given by

$$\Phi = \operatorname{atan}\left(\frac{YT_{aim} - YT_{fire}}{XT_{aim} - XT_{fire}}\right) \qquad \text{Eq. 7}$$

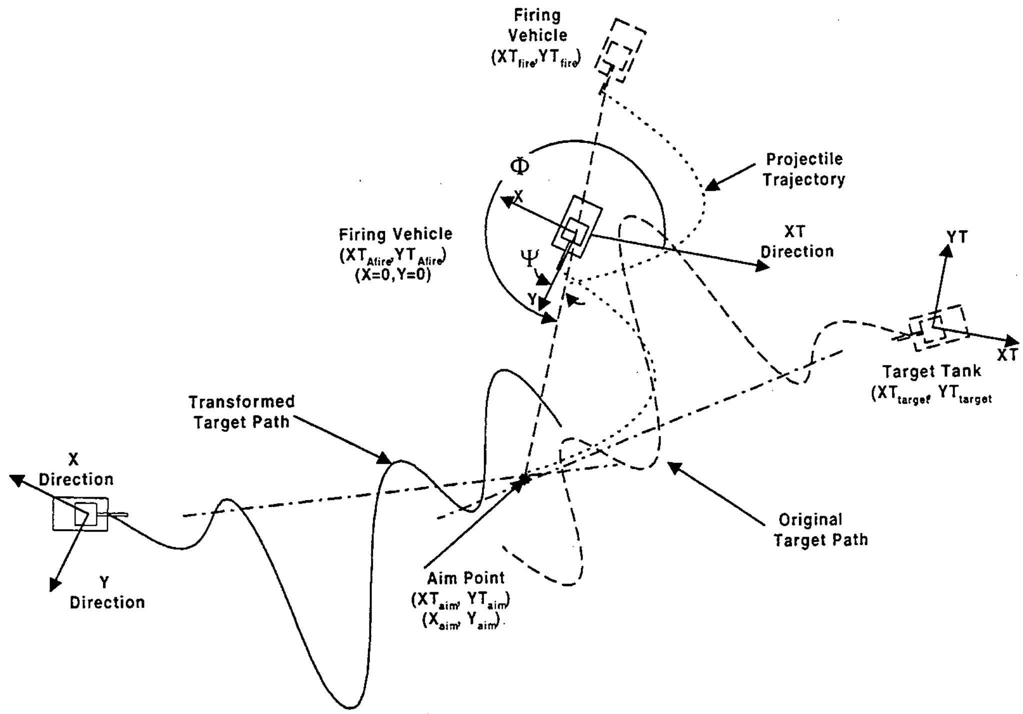

Figure 4. Transformation From Target Path to Trajectory Coordinates.

The length of the vector is also calculated; it is compared to the length of the vector from the firing tank to the end of the trajectory in *trajectory coordinates*. Because the trajectory file and the firing tank position inputs are not coordinated, the lengths of the two vectors may not be equal. The direction of the vector in *target path* coordinates is preserved, but the length is adjusted to make it equal to the length of the vector in trajectory coordinates. The length is adjusted by moving the firing tank position to (XT_{Afire}, YT_{Afire}).

The angle, in trajectory coordinates, between the Y axis and the vector connecting the firing tank to the end of the trajectory is given by the following:

$$\psi = \mathrm{atan}\left(\frac{X_{aim}}{Y_{aim}}\right) \qquad \text{Eq. 8}$$

The angle between the *target path* and the trajectory X direction is the following:

$$\xi = \left(\Phi - \frac{\pi}{2}\right) - \psi \qquad \text{Eq. 9}$$

To transform from *target path* coordinates into *trajectory coordinates*, they must be rotated about the Z axis by ξ. The translation moves the firing tank position from XA_{fire}, YA_{fire} to 0,0. The equation for the transformation is the following:

$$X_{target} = \left(XT_{target} - XT_{Afire}\right) \cdot \cos(\xi) + \left(YT_{target} - YT_{Afire}\right) \cdot \sin(\xi)$$

$$Y_{target} = \left(XT_{target} - XT_{Afire}\right) \cdot -\sin(\xi) + \left(YT_{target} - YT_{Afire}\right) \cdot \cos(\xi) \qquad \text{Eq. 10}$$

In Equation 10, XT_{target}, YT_{target} are coordinates in the *target path* system, and X_{target}, Y_{target} are coordinates in the *trajectory system.*

5. ANIMATION OF ENCOUNTERS

SWEM gives the user the option of viewing simple animations of the encounters between the weapon and the target. The animations show the encounters during each replication. Because the animations are computationally intensive, they greatly slow the calculation of the encounters and should not be used during long runs to collect statistics.

The graphical display shows the location of the projectile, the location of the target, and the footprints of the sensor(s) carried by the projectile. The center of the projectile is displayed as a small, blue, bullet-shaped icon. The target is displayed as a small, red, tank-shaped icon. These icons are meant to display only position; the orientation of the icon is not related to the orientation of the target or projectile.

As many as three sensors can be modeled, and the footprint of each is assigned a separate color. When the footprints of distinct sensors overlap, the colors are blended. For example, a blue footprint and a red footprint produce a purple overlap zone. Figure 5 shows an example of one frame of an encounter animation.

The center of the animation display is fixed in space and is located at the aim point of the projectile. The size the display is fixed at 1000 by 1000 meters. The display area is scaled to match the number of pixels available in the PC display. The target, whether stationary or moving, normally stays within the display area. The projectile and sensor footprints start outside the display area and do not appear until they move into the area in the course of the trajectory.

The footprint is displayed by applying Equations 5 and 6 to each of the pixels in the PC's displays. To reduce run time, only the pixels ahead of the projectile and within the maximum range of the sensor are verified. The footprint is animated by examining the pixels at every time step. As the footprint moves over pixels, they are colored. As the footprint moves off pixels, they return to the background color. The algorithm animates the sensor footprint's movement and its shape changes.

6. ARCHITECTURE OF SWEM CODE

Both the PC and UNIX™ versions of the SWEM code are written in C++. The PC version was developed from the UNIX™ version by adding a Windows™-based graphical user interface (GUI). Since it includes the UNIX™ code, the PC version is described in this report.

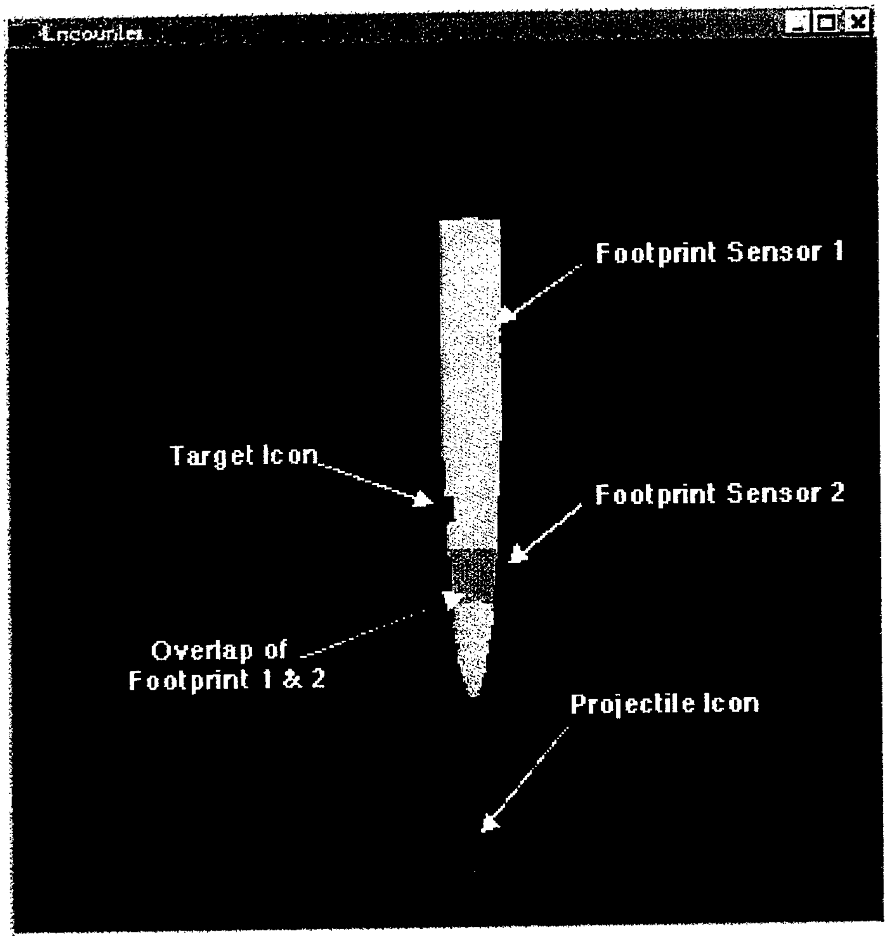

Figure 5. One Frame of the Encounter Animation.

The code includes 15 C++ classes and a "main" distributed in 18 files. Six of the classes concern the geometry and animation of the encounter described in Sections 2 through 5. These six classes contain approximately 2000 lines of code. The remaining nine classes comprise the GUI and contain about 4900 lines of code.

The SWEM code has a mixed architecture. The nine classes comprising the GUI are written with object-oriented methods. The six classes concerning the encounter's geometry and animation are not written with true object-oriented methods but with more traditional structured coding methods.

Figure 6 is flow chart showing the code's basic architecture in terms of traditional structured coding methods. The code consists of the GUI and three nested "for" loops. The GUI manages the user's input to the code. The outermost loop controls the statistical replications of the encounter simulations. The middle loop increments the time steps during an encounter. The innermost loop is actually two nested loops, one marching in X and the other marching in Y. The two loops increment through all the pixel locations during each time step to display the sensor's footprint.

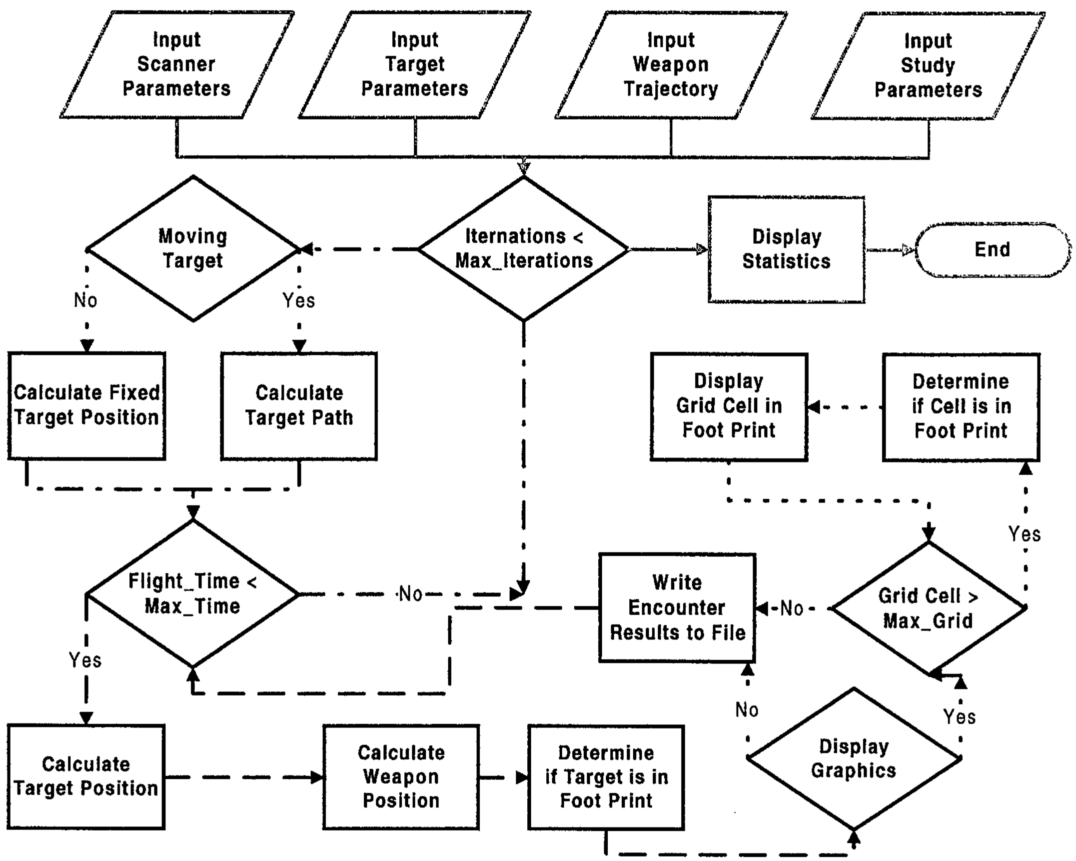

Figure 6. Flow Chart for the SWEM Code.

7. USER INPUT AND GRAPHICAL USER INTERFACE

User input to the SWEM code is done using the GUI and two standard input files. One of files contains the trajectory data for the projectile. The other file contains the data used by the "*makpth*" code to create the target path. The GUI uses five separate windows to input the data describing the weapon, the sensors, and the location of either stationary or moving targets.

SWEM can read the trajectory files in four different formats. Three of the formats are associated with specific TERM designs and are not covered in this report. An example of the fourth, or generic weapon format is given in Appendix A. The generic weapon file contains nine columns of data. The first column contains the trajectory time; the time in this column does not need to be evenly spaced. The second, third, and fourth columns contain the x, y, and z components of the projectile's location measured in meters. In the data format, X is in the down-range direction, Y is in the cross-range direction, and Z is in the local vertical direction. The fifth, sixth, and seventh columns contain the x, y, and z components of the projectile's velocity in meters per second. The last two columns contain the projectile's pitch and yaw angles in radians. The nine columns are separated by commas to ease importation into spreadsheets and plotting software.

An example of the input file for the *makpth* code is given in Appendix B. The first number in the file is the minimum duration of the target path. The first number on the second line is the mean of the X starting positions for the target. The second number is the standard deviation of the X starting positions. The third and fourth numbers are, respectively, the mean of the Y starting position and the standard deviation of the Y starting position. The first and second numbers on the third line are the mean and standard deviation of the target speed. The third and fourth numbers are the minimum and maximum target speed. The first and second numbers on the fourth line are the mean and standard deviation of the leg duration. The last two numbers are the minimum and maximum leg durations. The first number on the fifth line is the standard deviation for the target path angle; the mean is assumed to be 0 radians or east. The last two numbers are the minimum and maximum path angles in radians. The number on the sixth line indicates the tendency of the target path to return to its initial easterly direction. The value should be between 0 and 1. Entering a value of 0 produces no tendency to return to an easterly heading, whereas entering a 1 produces a strong tendency to do so.

Once the target path is calculated for a distribution about an easterly compass heading via the methods given, it is rotated as a whole through some angle picked from a random distribution. The first number on the seventh line determines which random number distribution is used to pick an overall path rotation angle. If the number is 1, a normal distribution is used. Otherwise, a uniform distribution is used. The second and third numbers are the mean and standard deviation used with the normal distribution of the rotation angle. The fourth and fifth numbers are the minimum and maximum rotation angles used with both the normal and uniform distributions.

The main "encounter" window in the GUI is shown in Figure 7. The first window is used to move to other windows in the GUI and to enter some basic information about the simulation. The "weapon platform parameters," "sensor parameters," "moving target," and "stationary target windows" are all accessed from the first window. The "reset to default" button in the upper right-hand corner allows the user to reinitialize all the parameters to a default set of values for testing and debugging the program.

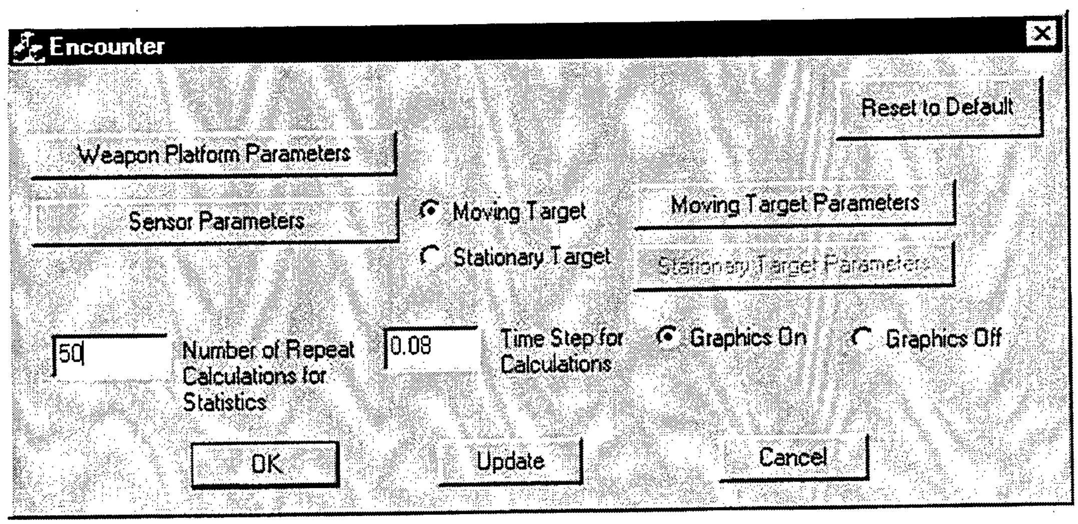

Figure 7. Encounter Window.

The selection of a "moving target" or "stationary target" is done using radio buttons. With radio buttons, indicated by small circles, selecting one automatically deselects the other(s). The text box labeled "number of repeat calculations for statistics" determines the number of replications of the simulation. The text box labeled "time step for calculations" sets the time in seconds between encounter checks and updates the animations. The "graphics on" and "graphics off" radio buttons determine if the animation is displayed. If five or more replications are used, the "graphics off" radio button should be pressed to prevent the run time from becoming too long.

The buttons at the bottom indicate what to do with the parameters entered into the GUI. The "OK" button causes the parameters to be saved as the current working set and the simulation to execute. The current set of parameters is saved to a hidden file and is read the next time the GUI is executed. The GUI always starts with the last set of parameters entered into SWEM. The "update" button causes the parameters to be saved to the hidden file as the current working set but does not execute the actual simulation. If the "cancel" button is pressed, the hidden file is not written nor is the simulation executed. The parameters are reset to the last working set.

When the "OK" button is pressed and the "graphics on" radio button is active, the animation window appears. Using the mouse to place the cursor in the animation window and right clicking causes another window to appear. The new window contains "begin" and "quit". Selecting "begin" starts the simulation and selecting "quit" aborts the simulation.

The "weapon platform parameters" window sets the number of sensors on the weapon, the sensor mounting method and the trajectory that the weapon flies. Figure 8 shows an example of the "weapon platform parameters" window. The four radio buttons at the top determine which of the trajectory file formats are used to read the data. Only the "generic weapon" format is covered in this report. An example of the "generic weapons" format is given in Appendix A. The user may specify a trajectory file directly using the text box labeled "trajectory filename". The user can also push the "browse" button and search for the proper file.

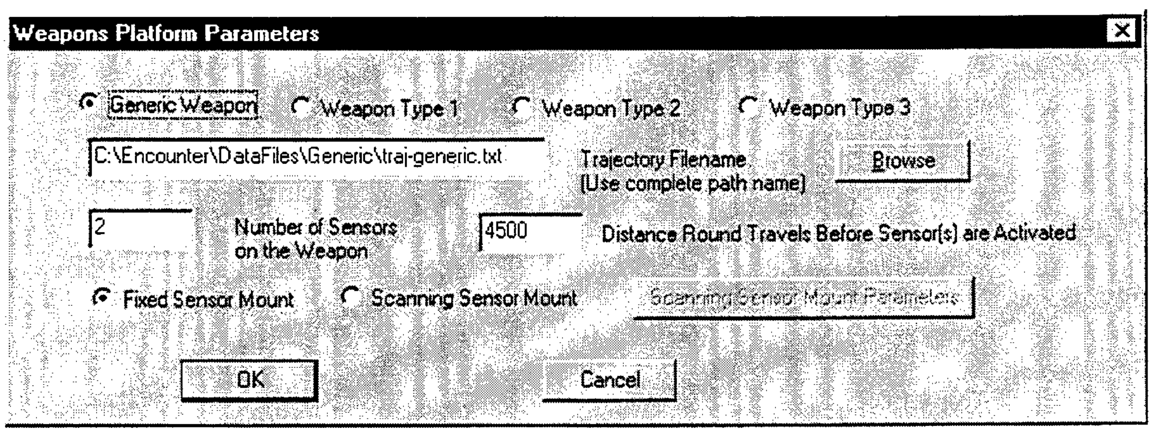

Figure 8. Weapons Platform Parameters Windows.

The number of sensors carried by the weapon is specified with the text box labeled "number of sensors on the weapon." The text box will not accept numbers larger than 3. The sensors on the weapon are not active at launch. The distance the weapon travels before the sensors activate is entered via the text box labeled "distance round travels before sensor(s) are activated." The user can specify either fixed or scanning mounts for the sensors by using the two radio buttons at the bottom of the window. If a scanning mount is selected, an additional button labeled "scanning sensor mount parameters" becomes available.

The "sensor parameters" window, accessible from the main window, allows the user to input the parameters describing as many as three sensors. The number of sensors specified in the "weapons platform parameters" window determines how many sets of text boxes are active in this window. Figure 9 shows the case when two sensors are specified. In this case, the text boxes on the top two sets are active and the lowest set is inactive.

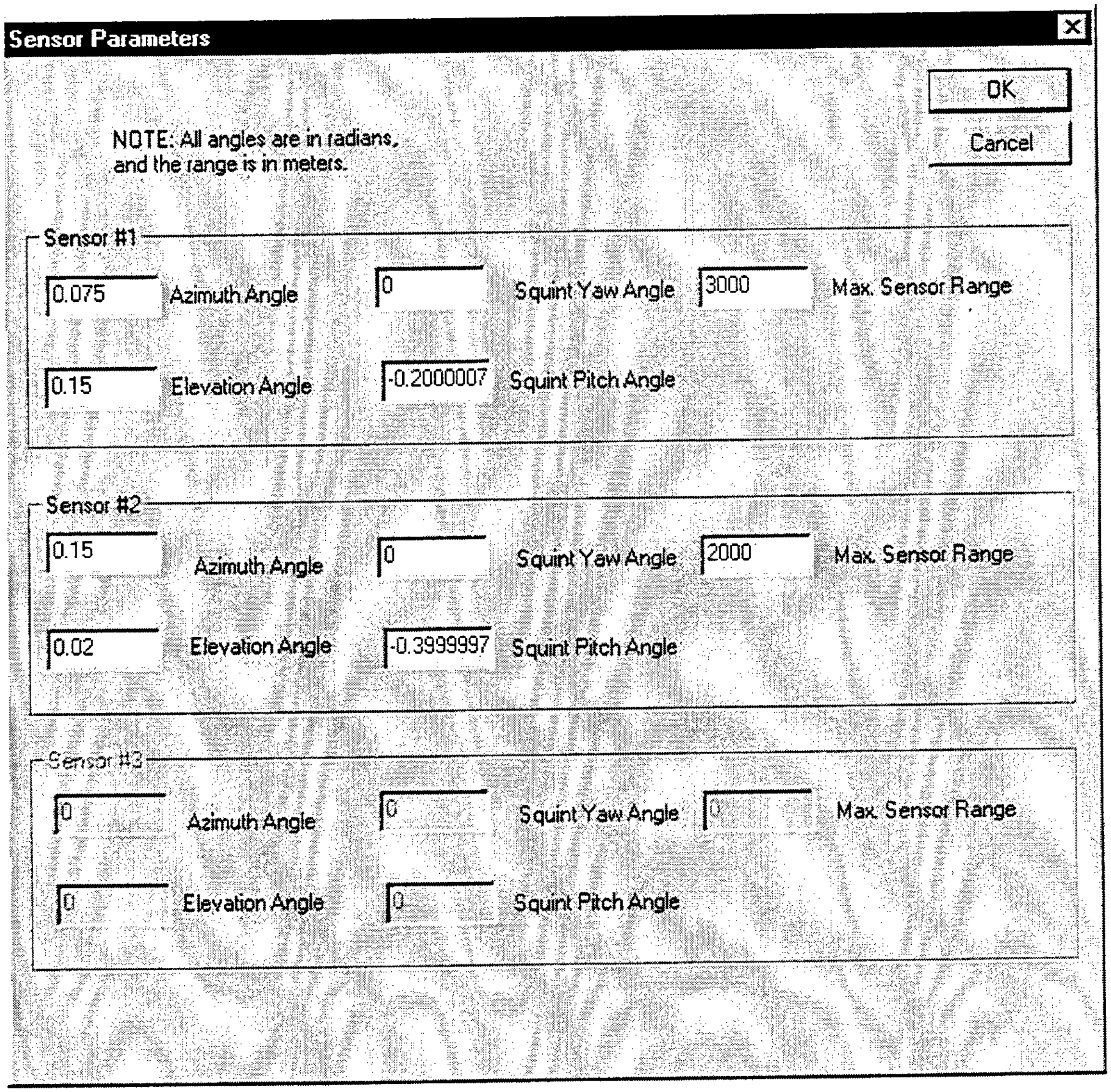

Figure 9. Sensor Parameter Window.

The text box labeled "azimuth angle" specifies the half angle in the horizontal plane of the FOV when the weapon is flying level. The text box labeled "elevation angle" specifies the half angle of the FOV in the vertical plane. The greatest range at which a sensor can detect a target is specified via the text box labeled "maximum sensor range." The text boxes labeled "squint yaw angle" and "squint pitch angle" are used to input the angular offsets of the sensor axis from the long axis of the weapon. The squint yaw angle is the offset in the horizontal plane when the weapon is flying level. The squint pitch angle is the offset in the vertical plane. For sensors in scanning mounts, the squint angles are the angles at activation, which change as the mount moves relative to the weapon. Sensors with fixed mounts maintain the same offset throughout the simulation. As stated in the note at the top of the window, all angles are input in radians and the maximum sensor ranges are in meters.

If the stationary target radio button is active in the main window, then the "stationary target parameters" window can be accessed. This window, shown in Figure 10, allows the user to input the parameter pertaining to the target location relative to the weapon's aim point as described in Section 3. The top four text boxes are used to input the TLE. There are text boxes for the mean and standard deviation in both range and deflection. The bottom four text boxes are used to input the WDE. Again, there are text boxes for the mean and standard deviation in both range and deflection.

In some TERM design concepts, the weapon is not initially aimed at the point where the target is calculated to be but rather at some point offset from it. The middle three text boxes are used to input the aim point offset if one is required. The aim point offset distances, like all other length parameters in this window, are given in meters.

If the "moving target" radio button is active in the main window, then the "moving target parameters" window can be accessed. This window, shown in Figure 11, allows the user to input the parameters pertaining to a moving target as described in Section 4.

The top two text boxes allow the user to specify the starting location of the target relative to the firing tank; this is equivalent to specifying the firing tank's location in *tank path coordinates*. The location is given in terms of range and deflection in meters.

The next three text boxes specify the timing of the encounter situation. The text box labeled "mean of randomly selected time of first observation" is used to input the mean of the distribution from which this time is picked. The text box labeled "standard deviation of randomly selected time of first observation" is used to input the required standard deviation. The text box labeled "time between first and second observations" allows the user to enter the time between the two observations of the target by the scout. The text box on the third row labeled "communication, decision, and implementation time" is used to enter the time between the second observation and the launch of the projectile. All the times in the first three rows are in seconds.

The bottom three rows of text boxes are the same as the text boxes in the "stationary target parameters." In the moving target case, these parameters are used to displace the point on the target path that corresponds to projectile impact time.

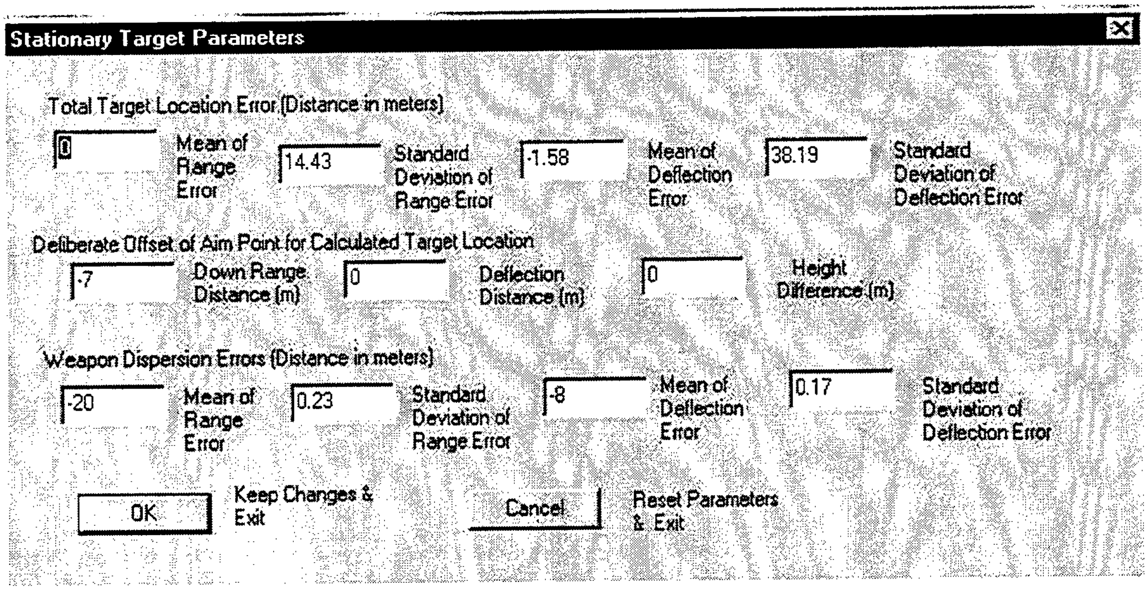

Figure 10. Stationary Target Parameter Window.

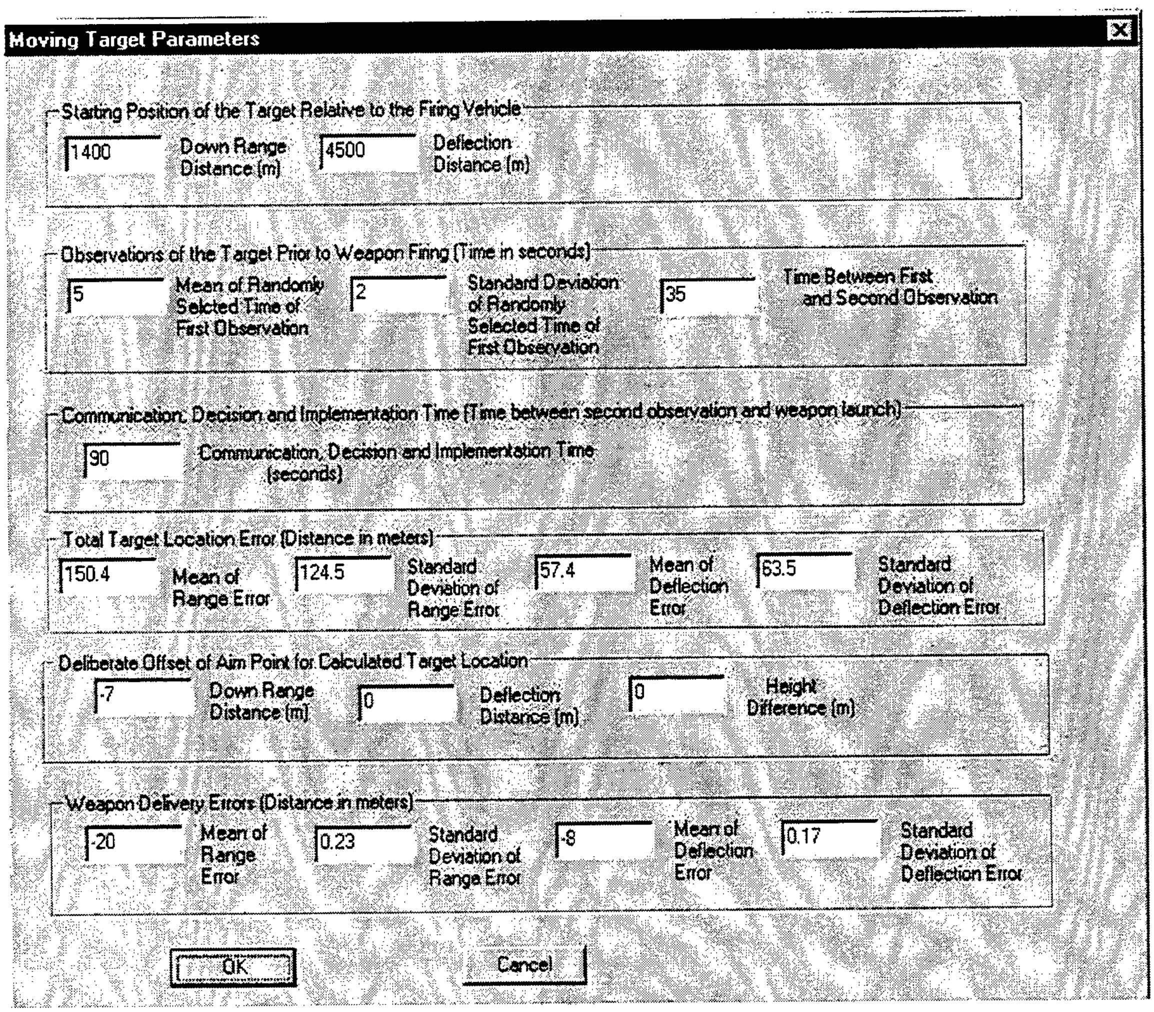

Figure 11. Moving Target Parameter Window.

If the "scanning sensor mount" radio button in the "scanning sensor mount parameters" window is active, then the "movement of gimbaled sensor mount" window can be accessed. This is the first window in a series of three windows used to input the movement of a gimbaled sensor mount. The mount movement algorithm assumes that the gimbaled mount moves only in yaw and starts at the yaw squint angle relative to the projectile's long axis. The mount can move through a series of scans. During each scan, it moves to a number of angular locations where it stops or dwells. At the end of the scan, the mount returns to the squint angle. Between dwell locations, the mount is assumed to move at a constant angular rate. At the end of a scan, the mount also returns to the dwell angle at the same rate. The dwell angular location and the length of time the mount spends at each dwell location are input values.

The "movement of gimbaled sensor mount" window is shown in Figure 12. The text box labeled "maximum gimbal rate" is used to input the angular velocity between dwell locations and upon return to the yaw squint angle. The number of scans is entered via the other text box.

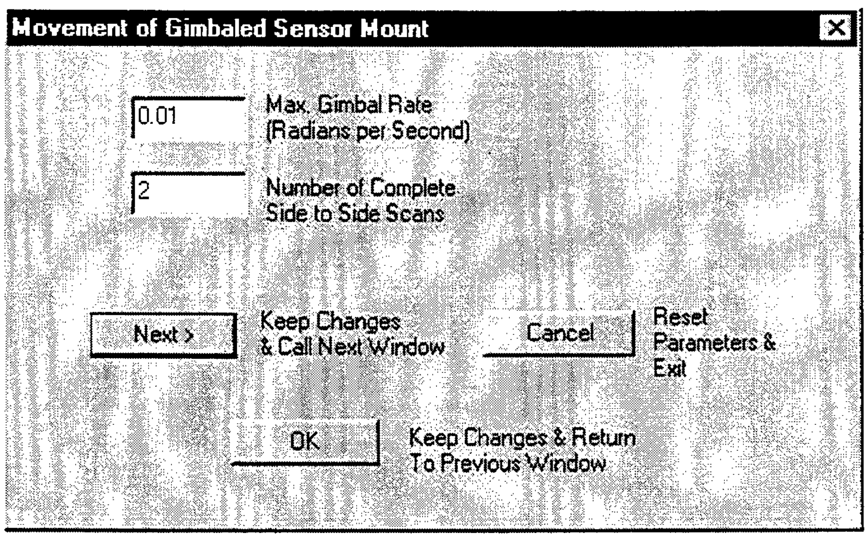

Figure 12. Movement of Gimbaled Sensor Mount Window.

When the "next" button is pressed in the "movement of gimbaled sensor mount" window, the "movement of gimbaled sensor mount during scan number" window appears. An example of this window is shown in Figure 13. The number of the scan for which the window is currently accepting data is given by the text box labeled "movement of gimbaled sensor mount during scan number." The number of dwell locations during the current scan is input into the text box in this window. Pressing the "previous" button returns the user to the "movement of gimbaled sensor mount" window. Pressing the "next" button causes the "parameters for dwell number" window to appear.

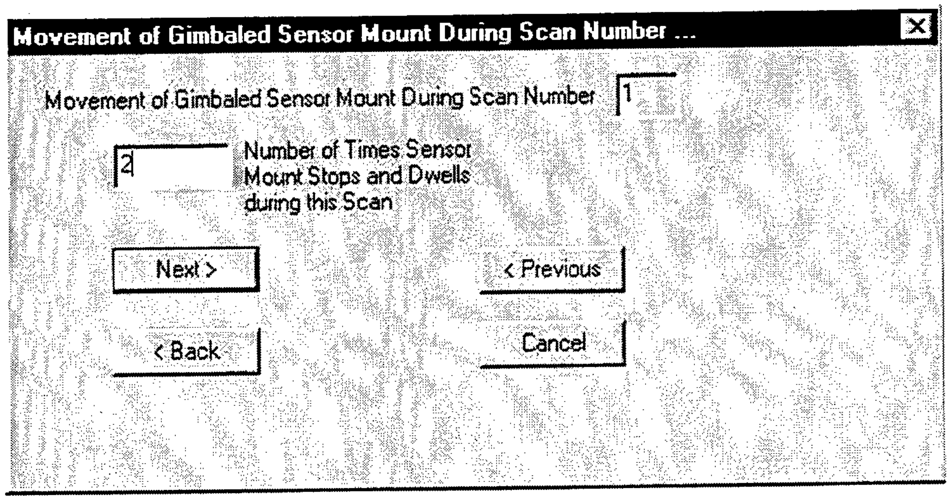

Figure 13. Movement of Gimbaled Sensor Mount During Scan Number Window.

The "parameters for dwell number" window is shown in Figure 14. The text box labeled "parameters for dwell number" indicates the current dwell number in the scan. The text box labeled "during scan number" indicates the current scan number. The text box labeled "yaw angle of mount" is used to input the location of the current dwell in radians from the long axis of the projectile. The text box labeled "duration of the dwell" is used to input the length of time (in seconds) the mount spends at the dwell angle after reaching that location. The amount of time it takes to move between dwell locations is determined by the maximum angular velocity of the mount and the angular distance between adjacent dwell locations.

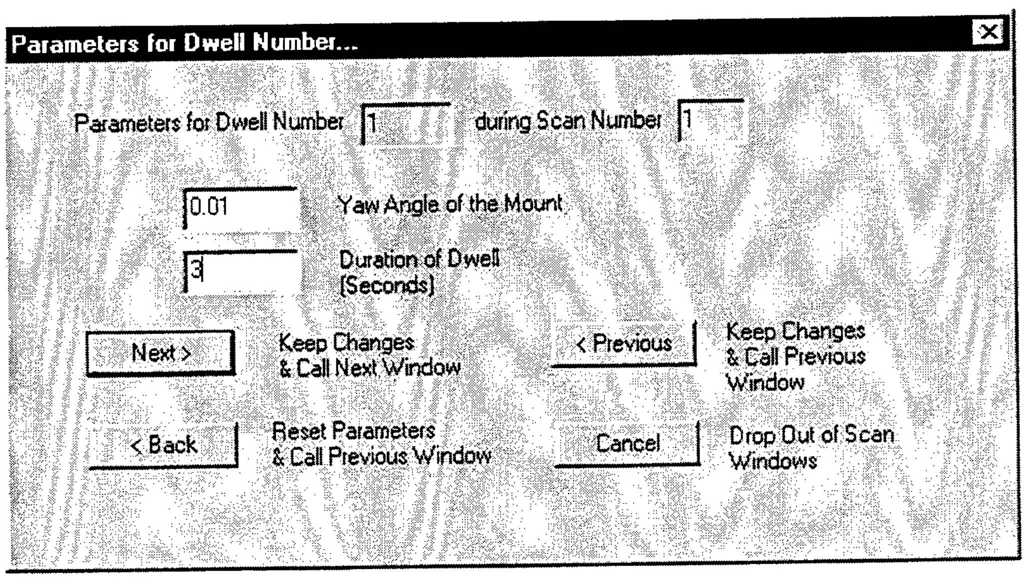

Figure 14. The Parameter of Dwell Number Window.

If the current scan in the "parameters for dwell number" is one, then pressing the "previous" button returns the user to the "movement of gimbaled sensor mount during scan number" window. If the current scan is not one, then pressing the "previous" button returns the user to the "parameters for dwell number" window for the preceding dwell. If the current dwell is not the last dwell in the scan, then pressing the "next" button calls a "parameters for dwell number" window for the next dwell. If it is the last dwell in the current scan but not the last dwell of the last scan, then the "next" button calls the "movement of gimbaled sensor mount during scan number" window. If the next dwell is the last dwell of the last scan, then pressing the "next" button calls the "movement of gimbaled sensor mount" window. Pressing the "OK" button in the "movement of gimbaled sensor mount" window ends the gimbaled mount input sequence.

8. STATISTICAL RESULTS FILE

The graphical output of the SWEM code has already been described in Section 5, so this section concentrates on the "encounter.txt" file that contains the statistical results after the program has been run. To operate properly, the SWEM code must be installed in a directory "C:\Encouner." The input files are located in "C:\encounter\DataFiles" and the output files in "C:\encounter\Output." The path to the statistical results is then in "C:\encounter\Output\encounter.txt." An example of the "encounter.txt" file is given in Appendix C. The file contains an entry for each replication and a summary of all the replications. If a sensor encounters a target during a particular replication, the file will list the duration of the encounter in seconds. The file will also contain the minimum and maximum range from the projectile to the target during the encounter. If no encounter occurs for a sensor during a replication, the file will contain a comment to that effect.

A summary of all the replications is given at the end of the "encounter.txt" file. For each sensor, the percentage of the replications for which there was an encounter is listed. These percentage values are the probability of encounter for that particular sensor. The last line gives the percentage of replications for which there was an encounter by any sensor on the weapon. This last value is the probability of encounter for the weapon as a whole.

9. CONCLUSIONS

The SWEM code was developed as an analytical tool for the TERM program. The SWEM code has been used to help evaluate TERM design concepts and has proved to be very useful. While the code is written for a cannon-launched smart projectile, it is general enough that it could be used in the analysis of other smart weapons designs.

The algorithm currently only considers flat terrain, but it could be extended to include the effects of terrain masking. The code could be modified to work with variable resolution terrain (Wald & Patterson 1992) or other terrain description formats. The code could also be adapted to simulate fragmentation weapons by representing the fragment pattern in a manner similar to the sensor's FOV. Neither of these extensions of the code is currently planned but would be relatively straightforward, given sufficient interest on the part of potential users.

REFERENCES

Meirovitch, L. "Methods of Analytical Dynamics," page 142. McGraw Hill Book Company, NY, 1970.

Wald, J.K., "Target Location Error Methodology for the Tank Extended Range Munition," ARL-TR-1433, U.S. Army Research laboratory, Aberdeen Proving Ground, MD 21005-5066, September 1997.

Wald, J.K., private communication, 1999.

Wald, J.K., and C.J. Patterson, "A Variable Resolution Terrain Model for Combat Simulation," Ballistic Research Laboratory, Aberdeen Proving Ground, MD 21005-5066, July 1992.

INTENTIONALLY LEFT BLANK

APPENDIX A

TRAJECTORY DATA FILE

INTENTIONALLY LEFT BLANK

Time	X	Y	Z	X_Dot	Y_Dot	Z_Dot	Yaw	Pitch
0.0528978,	13.7393,	7.5915e-05,	7.91867,	259.659,	-0.0983497,	149.395,	-0.000378765,	0.521273
0.105796,	27.4707,	-0.00504934,	15.8054,	259.51,	-0.196567,	148.791,	-0.000757454,	0.519774
0.158693,	41.1943,	-0.0153686,	23.6601,	259.361,	-0.294681,	148.187,	-0.00113618,	0.518272
0.211591,	54.91,	-0.0308763,	31.4829,	259.213,	-0.392693,	147.584,	-0.00151494,	0.516765
0.264489,	68.6178,	-0.0515672,	39.2739,	259.064,	-0.490603,	146.981,	-0.00189375,	0.515255
0.317387,	82.3178,	-0.0774358,	47.0329,	258.916,	-0.58841,	146.378,	-0.00227259,	0.513742
0.370284,	96.01,	-0.108477,	54.76,	258.767,	-0.686114,	145.775,	-0.00265146,	0.512225
0.423182,	109.694,	-0.144684,	62.4553,	258.619,	-0.783716,	145.173,	-0.00303038,	0.510705
0.47608,	123.371,	-0.186053,	70.1187,	258.471,	-0.881215,	144.572,	-0.00340933,	0.509181
0.528978,	137.039,	-0.232579,	77.7503,	258.323,	-0.978612,	143.97,	-0.00378831,	0.507653
0.581876,	150.7,	-0.284254,	85.3501,	258.175,	-1.07591,	143.369,	-0.00416733,	0.506122
0.634773,	164.353,	-0.341075,	92.9181,	258.027,	-1.1731,	142.768,	-0.00454639,	0.504587
0.687671,	177.998,	-0.403036,	100.454,	257.879,	-1.27019,	142.168,	-0.00492548,	0.503049
0.740569,	191.635,	-0.47013,	107.959,	257.731,	-1.36717,	141.568,	-0.0053046,	0.501508
0.793467,	205.265,	-0.542354,	115.432,	257.583,	-1.46405,	140.968,	-0.00568376,	0.499962
0.846365,	218.887,	-0.619701,	122.873,	257.435,	-1.56084,	140.368,	-0.00606295,	0.498414
0.899262,	232.5,	-0.702166,	130.282,	257.287,	-1.65751,	139.769,	-0.00644218,	0.496861
0.95216,	246.106,	-0.789744,	137.66,	257.14,	-1.75409,	139.17,	-0.00682144,	0.495305
1.00506,	259.705,	-0.882429,	145.006,	256.992,	-1.85056,	138.572,	-0.00720073,	0.493745
1.05796,	273.295,	-0.980216,	152.32,	256.845,	-1.94693,	137.974,	-0.00758005,	0.492182
1.11085,	286.878,	-1.0831,	159.603,	256.697,	-2.0432,	137.376,	-0.0079594,	0.490616
1.16375,	300.452,	-1.19107,	166.854,	256.55,	-2.13936,	136.779,	-0.00833879,	0.489045
1.21665,	314.02,	-1.30413,	174.073,	256.403,	-2.23543,	136.182,	-0.00871821,	0.487471
1.26955,	327.579,	-1.42227,	181.261,	256.255,	-2.33139,	135.585,	-0.00909766,	0.485894
1.32244,	341.13,	-1.54548,	188.417,	256.108,	-2.42724,	134.988,	-0.00947713,	0.484313
1.37534,	354.674,	-1.67377,	195.542,	255.961,	-2.523,	134.392,	-0.00985664,	0.482728
1.42824,	368.21,	-1.80711,	202.636,	255.814,	-2.61865,	133.797,	-0.0102362,	0.48114
1.48114,	381.738,	-1.94552,	209.697,	255.667,	-2.7142,	133.201,	-0.0106157,	0.479548
1.53404,	395.258,	-2.08898,	216.728,	255.52,	-2.80965,	132.606,	-0.0109953,	0.477952
1.58693,	408.771,	-2.23748,	223.726,	255.374,	-2.90499,	132.011,	-0.011375,	0.476353
1.63983,	422.276,	-2.39103,	230.694,	255.227,	-3.00024,	131.417,	-0.0117546,	0.47475
1.69273,	435.773,	-2.54961,	237.63,	255.08,	-3.09538,	130.823,	-0.0121343,	0.473144
*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	*	*

27.4011,	6247.02,	-589.552,	215.766,	200.725,	-39.7522,	-121.89,	-0.195513,	-0.537622
27.454,	6257.63,	-591.654,	209.306,	200.62,	-39.8086,	-122.345,	-0.195883,	-0.539462
27.5069,	6268.24,	-593.76,	202.823,	200.515,	-39.8648,	-122.8,	-0.196253,	-0.541299
27.5598,	6278.85,	-595.868,	196.315,	200.411,	-39.921,	-123.254,	-0.196622,	-0.543132
27.6127,	6289.45,	-597.979,	189.783,	200.306,	-39.9771,	-123.708,	-0.196992,	-0.544962
27.6656,	6300.04,	-600.093,	183.227,	200.201,	-40.033,	-124.162,	-0.197361,	-0.546788
27.7185,	6310.63,	-602.21,	176.647,	200.096,	-40.0889,	-124.615,	-0.197731,	-0.548612
27.7714,	6321.21,	-604.33,	170.043,	199.991,	-40.1446,	-125.069,	-0.1981,	-0.550431
27.8243,	6331.78,	-606.453,	163.415,	199.885,	-40.2003,	-125.521,	-0.198469,	-0.552248
27.8772,	6342.35,	-608.579,	156.764,	199.78,	-40.2558,	-125.974,	-0.198838,	-0.55406
27.9301,	6352.92,	-610.708,	150.088,	199.675,	-40.3112,	-126.426,	-0.199207,	-0.55587
27.983,	6363.48,	-612.839,	143.388,	199.569,	-40.3666,	-126.878,	-0.199576,	-0.557676
28.0359,	6374.03,	-614.974,	136.665,	199.464,	-40.4218,	-127.329,	-0.199945,	-0.559479
28.0888,	6384.58,	-617.112,	129.917,	199.358,	-40.4769,	-127.78,	-0.200314,	-0.561278
28.1417,	6395.12,	-619.252,	123.146,	199.252,	-40.5319,	-128.231,	-0.200682,	-0.563074
28.1946,	6405.66,	-621.396,	116.351,	199.146,	-40.5869,	-128.682,	-0.201051,	-0.564866
28.2475,	6416.19,	-623.542,	109.532,	199.041,	-40.6417,	-129.132,	-0.201419,	-0.566656
28.3004,	6426.72,	-625.691,	102.69,	198.935,	-40.6964,	-129.582,	-0.201787,	-0.568441
28.3533,	6437.24,	-627.843,	95.8231,	198.828,	-40.751,	-130.031,	-0.202156,	-0.570224
28.4062,	6447.75,	-629.998,	88.9329,	198.722,	-40.8054,	-130.48,	-0.202524,	-0.572002
28.4591,	6458.26,	-632.156,	82.0189,	198.616,	-40.8598,	-130.929,	-0.202892,	-0.573778
28.512,	6468.77,	-634.317,	75.0812,	198.51,	-40.9141,	-131.378,	-0.20326,	-0.57555
28.5649,	6479.26,	-636.48,	68.1197,	198.403,	-40.9683,	-131.826,	-0.203628,	-0.577319
28.6178,	6489.76,	-638.647,	61.1346,	198.297,	-41.0223,	-132.274,	-0.203996,	-0.579084
28.6707,	6500.24,	-640.816,	54.1257,	198.19,	-41.0763,	-132.721,	-0.204363,	-0.580846
28.7236,	6510.72,	-642.988,	47.0932,	198.084,	-41.1301,	-133.168,	-0.204731,	-0.582604
28.7765,	6521.2,	-645.163,	40.0371,	197.977,	-41.1838,	-133.615,	-0.205098,	-0.584359
28.8294,	6531.67,	-647.341,	32.9573,	197.87,	-41.2374,	-134.062,	-0.205466,	-0.586111
28.8823,	6542.13,	-649.522,	25.854,	197.763,	-41.291,	-134.508,	-0.205833,	-0.587859
28.9352,	6552.59,	-651.706,	18.727,	197.656,	-41.3444,	-134.954,	-0.2062,	-0.589604
28.9881,	6563.04,	-653.892,	11.5765,	197.549,	-41.3977,	-135.399,	-0.206567,	-0.591345
29.041,	6573.49,	-656.081,	4.40237,	197.442,	-41.4508,	-135.844,	-0.206934,	-0.593083
29.0939,	6583.93,	-658.273,	-2.79526,	197.334,	-41.5039,	-136.289,	-0.207301,	-0.594817

APPENDIX B

INPUT TO THE MAKPTH CODE

INTENTIONALLY LEFT BLANK

INPUT TO THE MAKPTH CODE

```
240.0
     0.00   200.00        0.00     200.00
    12.50     5.00        5.00      20.00
    20.00     5.00       15.00      35.00
              0.34       -0.68       0.68
     1.00
1    1.57     0.01745     0.00       3.14
```

INTENTIONALLY LEFT BLANK

APPENDIX C

STATISTICAL RESULTS

INTENTIONALLY LEFT BLANK

STATISTICAL RESULTS

==================================

TEST RUN # 1

==================================

No Encounter by Sensor Number = 1

No Encounter by Sensor Number = 2

==================================

TEST RUN # 2

==================================

No Encounter by Sensor Number = 1

No Encounter by Sensor Number = 2

==================================

TEST RUN # 3

==================================

Sensor Number = 1
Duration of Encounter = 5.68
Minimum Range during Encounter = 985.024
Maximum Range during Encounter = 2689.84

Sensor Number = 2
Duration of Encounter = 5.36
Minimum Range during Encounter = 519.186
Maximum Range during Encounter = 1984.16

==================================

TEST RUN # 4

==================================

No Encounter by Sensor Number = 1

No Encounter by Sensor Number = 2

==================================

TEST RUN # 5

==================================

Sensor Number = 1
Duration of Encounter = 5.04
Minimum Range during Encounter = 1312.36
Maximum Range during Encounter = 2981.12

Sensor Number = 2
Duration of Encounter = 0.64

Minimum Range during Encounter = 437.966
Maximum Range during Encounter = 609.219

*
*
*
*
*
*
*

====================================
TEST RUN # 48
====================================

Sensor Number = 1
Duration of Encounter = 1.12
Minimum Range during Encounter = 2325.86
Maximum Range during Encounter = 2694.99

No Encounter by Sensor Number = 2
====================================
TEST RUN # 49
====================================

Sensor Number = 1
Duration of Encounter = 2.64
Minimum Range during Encounter = 2003.9
Maximum Range during Encounter = 2880.63

No Encounter by Sensor Number = 2
====================================
TEST RUN # 50
====================================

Sensor Number = 1
Duration of Encounter = 7.28
Minimum Range during Encounter = 320.343
Maximum Range during Encounter = 2394.7

Sensor Number = 2
Duration of Encounter = 3.44
Minimum Range during Encounter = 783.011
Maximum Range during Encounter = 1752.43

===
Final Totals
===

74 % of the tests had an encounter by sensor # 1
64 % of the tests had an encounter by sensor # 2

82 % of the tests had an encounter by any sensor

INTENTIONALLY LEFT BLANK

NO. OF COPIES	ORGANIZATION
2	ADMINISTRATOR DEFENSE TECHNICAL INFO CTR ATTN DTIC OCP 8725 JOHN J KINGMAN RD STE 0944 FT BELVOIR VA 22060-6218
1	DIRECTOR US ARMY RSCH LABORATORY ATTN AMSRL CS AL TA REC MGMT 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RSCH LABORATORY ATTN AMSRL CI LL TECH LIB 2800 POWDER MILL RD ADELPHI MD 207830-1197
1	DIRECTOR US ARMY RSCH LABORATORY ATTN AMSRL DD 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RSCH LABORATORY ATTN AMSRL D D R SMITH 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	OSD OUSD(A&T)/ODDDR&E(R) ATTN R J TREW THE PENTAGON WASHINGTON DC 20301-7100
1	US ARMY INFO SYS ENGRG CMD ATTN ASQB OTD F JENIA FT HUACHUCA AZ 85613-5300
1	AMCOM MRDEC ATTN AMSMI RD W C MCCORKLE REDSTONE ARSENAL AL 35898-5240
1	US ARMY TANK-AUTOMOTIVE CMD RD&E CTR ATTN AMSTA TA J CHAPIN WARREN MI 48397-5000
1	CECOM ATTN PM GPS COL S YOUNG FT MONMOUTH NJ 07703

NO. OF COPIES	ORGANIZATION
1	INST FOR ADVNCD TCHNLGY THE UNIV OF TEXAS AT AUSTIN PO BOX 20797 AUSTIN TX 78720-1714
1	CECOM SP & TERRESTRIAL COMMCTN DIV ATTN AMSEL RD ST MC M H SOICHER FT MONMOUTH NJ 07703
1	DARPA 3701 N FAIRFAX DR ARLINGTON VA 22203-1714
1	US ARMY NATICK RDEC ACTING TECHNICAL DIR ATTN SSCNC T P BRANDLER NATICK MA 01760-5002
1	NAVAL SURFACE WARFARE CTR CODE B07 J PENNELLA 17320 DAHLGREN RD BLDG 1470 RM 1101 DALGREN VA 224480-5100
1	US ARMY TRAINING & DOCTRINE CMD BATTLE INTEGRATION & TECH DIR ATTN ACTD B J A KLEVECZ FT MONROE VA 23651-5850
1	US MILITARY ACADEMY MATH SCI CTR OF EXCELLENCE DEPT OF MATHEMAICAL SCI MAJ MD PHILLIPS THAYER HALL WEST POINT NY 10996-1786
1	UNIV OF TEXAS ARL ELECTROMAG GROUP CAMPUS MAIL CODE F0250 ATTN A TUCKER AUSTIN TX 78713-8029
1	HICKS & ASSOCIATES, INC. ATTN G SINGLEY III 1710 GOODRICH DR STE 1300 MCLEAN VA 22102
2	NIST ATTN K MURPHY 100 BUREAU DRIVE GAITHERSBURG MD 20899

NO. OF COPIES	ORGANIZATION
1	SPECIAL ASST TO THE WING CDR 50SW/CCX CAPT P H BERSTEIN 300 O'MALLEY AVE STE 20 FALCON AFB CO 80912-3020
2	US ARMY MOUNTED MANEUVER BATTLELAB ATTN MAJ J BURNS BLDG 2021BLACKHORSE REG DR FORT KNOX KY 40121
1	HQ AFWA/DNX 106 PEACEKEEPER DR STE 2N3 OFFUTT AFB 68113-4039
2	NASA JET PROPULSION LAB ATTN L MATHIES K OWENS 4800 OAK GROVE DR PASADENA CA 91109
4	ROBOTICS SYS TECNOLOGY INC ATTN S MYERS P CORY B BEESON J ROBERTSON 1234 TECH COURT WESTMINISTER MD 21157
2	COMMANDER US ARMY TACOM ATTN J JACZKOWSKI WARREN MI 48397-5000
2	DIRECTOR US ARMY WATERWAYS EXP STN ATTN R AJLVIN 3909 HALLS FERRY ROAD VICKSBURG MS 39180-6199
1	US ARMY SIMULATION TRAIN & INSTRMNTN CMD ATTN J STAHL 12350 RESEARCH PARKWAY ORLANDO FL 32826-3726
	ABERDEEN PROVING GROUND
2	DIRECTOR US ARMY RSCH LABORATORY ATTN AMSRL CI LP (TECH LIB) BLDG 305 APG AA
1	US ARMY EDGEWOOD RDEC ATTN SCBRD TD J VERVIER APG MD 21010-5423
3	CDR USA DTC ATTN AMSTE CD B SIMMONS AMSTE CD R COZBY J HAUG RYAN BLDG
4	CDR USA ATC ATTN STEAC CO COL ELLIS STEAC TD J FASIG STEAC TE R CUNNINGHAM STEAC RM A MOORE BLDG 400
2	CDR USA ATC ATTN STEAC TE F P OXENBERG STEAC TE F A SCRAMLIN BLDG 321
1	DIR USARL ATTN AMSRL WM I MAY BLDG 4600
2	DIR USARL ATTN AMSRL WM B A W HORST JR W CIEPIELA BLDG 4600
1	DIR USARL ATTN AMSRL WM B E M SCHMIDT BLDG 390A
1	DIR ARL ATTN AMSRL WM BA W D'AMICO BLDG 4600
3	DIR USARL ATTN AMSRL WM BB H ROGERS G HAAS B HAUG BLDG 1121
1	DIR USARL ATTN AMSRL WM BC P PLOSTINS BLDG 390
1	DIR USARL ATTN AMSRL WM BD B FORCH BLDG 4600

NO. OF COPIES	ORGANIZATION
1	DIR USARL AMSRL WM BE G WREN BLDG 390
8	DIR USARL ATTN AMSRL WM BF J LACETERA P CORCORAN M FIELDS R PEARSON (5 CYS) BLDG 120
2	DIR USARL ATTN AMSRL WM BR C SHOEMAKER J BORNSTEIN BLDG 1121
	ABSTRACT ONLY
1	DIRECTOR US ARMY RSCH LABORATORY ATTN AMSRL CS AL TP TECH PUB BR 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	US ARMY RESEARCH OFC 4300 S MIAMI BLVD RESEARCH TRIANGLE PARK NC 27709
5	US ARMY TANK-AUTOMOTIVE & ARMAMENTS CMD ATTN AMSTA AR TD M FISETTE AMSTA AR FSS I (M BORTAK) AMSTA AR FSP S (R CASELE) AMSTA AR CC H A (J CHEW) AMSTA AR CCH B (P DORADIO) BLDG 1 PICATINNY ARSENAL NJ 07806-5000
1	DPTY CG FOR RDE US ARMY MATERIEL CMD ATTN AMCRD MG CALDWELL 5001 EISENHOWER AVE ALEXANDRIA VA 22333-0001

INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	April 2000	Final

4. TITLE AND SUBTITLE

Smart Weapons Encounter Model

5. FUNDING NUMBERS

PR: 1L162618AH80

6. AUTHOR(S)

Pearson, R.J.; Chien, K.K. (both of ARL)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

U.S. Army Research Laboratory
Weapons & Materials Research Directorate
Aberdeen Proving Ground, MD 21005-5066

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Research Laboratory
Weapons & Materials Research Directorate
Aberdeen Proving Ground, MD 21005-5066

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

ARL-TR-2178

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report covers the Smart Weapon Encounter Model (SWEM) developed to support the Tank Extended Range Munition (TERM) science and technology objective (STO) III G.3. The report decribes the model's algorithm, input, and output. SWEM uses solid geometry and statistical methods to calculate the probability that a target will fall within the field of view (FOV) of a sensor mounted on a smart weapon. When the target passes into the FOV of the sensor, an encounter is said to have occurred. SWEM calculates the probability of such encounters for a given sensor on a weapon traveling along a specific trajectory. The probability is determined by repeating a series of calculations in which the target's location or movements are randomly varied from calculation to calculation. Only the probability of encounter is determined. SWEM does not calculate the probability that the sensor system will be able to detect an encountered target against its background environment. Neither does it determine the probability that the weapon will be able to maneuver to a target once it is detected. However, SWEM does calculate the first necessary step in a series of steps leading to target interception.

14. SUBJECT TERMS

encounter	sensors	tank extended range munition
field of view	smart weapons	TERM

15. NUMBER OF PAGES

50

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102